华中科技大学

课 程 实 验 报 告

课程名称： Python 大数据与人工智能实践

实验名称： 电影评分预测

院　　系 ：计算机科学与技术学院

专业班级 ： 本硕博 202201 班

学　　号 ： U202215703

姓　　名 ： 张家万

指导教师 ： 邹逸雄

2023 年 12 月 29 日

一、任务说明

(1) 数据说明

采用Movie Lens Dataset，包含 9742 部电影、100836 个星级评分、610 位用户、3683 个标签。要求根据电影属性预测电影星级评分

(2) 基本内容

- 将 9742 部电影随机划分训练集（data_train）与验证集（data_val）。
- 使用训练集训练模型，预测验证集电影的标签。
- 重复三次“训练集/验证集划分、训练、测试”，汇报三次的平均准确率
- 数据集划分：

```
from sklearn.model_selection import train_test_split
data_train, data_val = train_test_split(data, test_size = 0.1)
```

(3) 任务提示

1. 数据读取：表格操作
2. 数据探索：每个属性与星级的关系，如何展示等
3. 数据预处理：缺失值、多用户等
4. 数据建模：除 csv 中现成信息外，电影海报等额外信息是否与星级有关?

鼓励利用 link.csv 链接爬虫获取额外信息
例如：https://www.imdb.com/title/tt0114709/ 中 114709 为 link.csv 中某一电影的 imbID

二、问题分析

(1)数据预处理

任务要求能根据电影的属性来预测电影的星级评分，即属性为模型的特征，评分为模型的标签。

后者可以根据 rating.csv 文件中各用户对电影的评分来得到，有两种处理方式，对于回归问题，直接用电影的平均评分作为标签，对于分类问题，可以将评分划分为(0,1]，(1,2]

，(2,3]， (3,4]，(4,5]这 5 个类别，分别以 0、1、2、3、4 作为标签，此操作可以利用 pandas.cut 的分桶操作来实现，也可以直接取平均分数的下界来实现。

对与前者，需要对给出的数据文件分别做处理，提取出有用的信息作为电影的属性。对于已有的数据的处理如下：

1. movies.csv，包含 movieId,tittle,genres。可以利用正则表达式从 tittle 中提取出电影名和上映年份，实验用到了上映年份作为特征之一，需要将其由字符串类型转为整型数据。对于 genres，表示电影的类别，可以采取独热编码，或是利用 MultiLabelBinarizer 生成是否属于某个类别的多个列的数据。
2. tags.csv，包含 userId, movieId, tag,timestamp，即用户对电影给出的 tag 标签和时间戳。可以利用 groupby 统计每部电影的所有 tag，虽然 tag 数据有 3683 条，但实际只有 1572 个电影有 tag, 且 tag 为文本信息，难以较好的处理。本次实验中只是统计了各电影的 tag 数'tag_count'，对于没有 tag 的用 0 代替。然后用一个很简单的 TextBlob 库的函数得到 tag 对应的情感计分值'Sentiment'，用这两项作为模型的特征
3. ratings.csv，包含 userId,movieId,rating,timestamp，即用户对电影给出的评分和时间戳。评分作为模型标签的处理前面已经说明，此外，统计了每部电影被给出评分的个数'rating_count'作为模型的特征之一。

考虑到原始数据集中方便作为特征用于模型训练的数据较少，可以通过 link.csv 中的信息爬取各电影在 imdb 网站上的信息。

本次实验爬取了电影的时长，imdb 星级、Users reviews 数，Critic reviews 数，计算电影导演的电影的平均星级。只有 3 个样本未能成功爬取（状态码 404），其中一个网站信息缺失严重，可以不管，另外两个通过百度搜索更新 imdb 号重新爬取。成功爬取的样本的电影时长和 imbd 星级无缺失值，Users reviews 和 Critic reviews，电影导演的电影的平均星级，有缺失的用 0 代替（网站本身无这些指标，应该是该电影还没有相应的 review，导演页面还未给出导演过的电影），自己爬取的信息放在了 info.csv 文件中。

此外还借用了别人已经爬好的信息，文件名为 film_ori.csv，包括电影的预算，三种票房（北美票房,首周票房、全球票房,缺失值可用均值或中位数代替,测试发现均值效果更好），和电影的 review 文本。对于 review 文本，可以使用 bert 模型和 lstm 转化为数值特征，但实验过程中发现加入 review 对应的数值数据后模型效果反而都变差了，而且处理时间大大增加，故不再添加该数据到初步预处理的数据中去。

然后将前面提到的各种特征和电影的评分合并起来，按前面提到的，count 有缺失的用 0 代替，票房有缺失的用中位数或是均值代替。然后丢掉不需要的属性，将作为模型标签的评分数据移到最后一列。输出有缺失值的各样本信息（可以手动搜索进行补充部分缺失信息），对于难以补充的含缺失值的信息，可以直接丢掉（实验中最后只有 19 条缺失，其中 1 条由于爬取信息缺失严重，其余 18 条没有 rating,不能作为模型训练的样本）

（2）模型搭建

对于爬虫代码，可以构建黑白名单登记已爬取和爬取异常的名单，这样可以保存爬取的进度，防止重复爬取或被迫从头开始爬取，构建 logging 记录爬取日志，方便处理异常优化爬虫代码。同时还可以开多线程来加速爬取速度。此前我也尝试过异步访问的方式，但要主语限制提交网络请求的速度，否则容易导致网站会暂时无法访问一段时间（大量样本的状态码为 503）。最选取多线程这一容易实现且较为稳定的方法，可以将爬取所需的时长由 13 个小时缩短为 2 个小时以内。此外，可以利用 concurrent.futures.as_completed 和 tqdm 来显示

爬取的进度条。

根据 rating 的不同处理，可以将问题分为回归问题和分类问题。为此我搭建了 6 个模型，分别为逻辑回归模型（实际是用于分类问题，对应 model1），随机森林分类模型（对应 model2 和 model2_optimized）、神经网络分类模型（对应 model3），线性回归模型（对应 model4）、随机森林回归模型（对应 model5 和 model5_optimized）、神经网络回归模型（对应 model6）。其中逻辑回归模型和线性回归模型分别为分类和回归问题的最简单的模型，可作为模型效果衡量的基准。

这些模型的公共部分是数据预处理，包括读取数据集，划分数据集为训练集和验证集（比例为 9：1），特征选取和归一化操作。对于神经网络模型，还需要转为 tensor 向量。此外，公共部分还有模型评估，有两种评估方式，对于分类模型，输出验证集上的准确率和分类报告（利用 classification_report）,对于回归模型，输出验证集上的 mse，rmse 和 accuray（其中 accuracy 用 1-mse 表示）。

对于两种随机森林模型，其共用特征重要性分析和贝叶斯优化调参的类或方法。特征重要性分析可以获取 model.feature_importances_和选取的特征列的列名，然后按一定格式输出模型训练后各特征的重要性，这在一定程度上也反应了各特征与电影星级评分的相关性。贝叶斯优化可以构建一个类，构造时传入的参数是 mode（表示是回归模型还是分类模型），model，x_train, x_valid, y_train, y_valid。初始化给出建立的树的个数 n_estimators，最大特征的选择方式 max_features，树的最大深度 max_depth，节点最小分裂所需要的样本数 min_samples_split 和叶子节点最小样本数 min_samples_leaf，用元组表示各自的搜索区间，然后用一个字典 search_grid 列出前面的这些参数表示搜索的参数空间。贝叶斯优化需要用到一个黑盒函数，其返回在给定参数下模型的效果作为贝叶斯优化的目标值 target，对于分类问题，返回验证集上的准确率，对于回归问题，返回 1-mse（注意贝叶斯优化是使得 target 越大越好）。然后用 BayesianOptimization 定义一个优化器 optimizer，参数 f 值为黑盒函数，pbounds 为参数搜索空间。然后调用 maximize 开始执行贝叶斯优化，其中参数 init_points 给出随机搜索的步数，n_iter 给出执行贝叶斯优化的步数，打印 optimizer.max 可以输出最佳参数和目标值。可以对比优化前和优化后两种随机森林模型的效果。

对于神经网络模型，我用到了三个 Linear 层，和两层 ReLU 激活函数，第一层输入的 input_dim 即为选取的特征数，最后一层的 output_dim 对于分类问题值为 5，对于回归问题值为 1。关于 loss 的选择，对于分类问题，选用 CrossEntropyLoss，对于回归问题，选用 MSELoss。模型的优化器采用 Adam 优化器。其余的部分比较模板化不再赘述，具体直接查看代码。值得注意的是模型用到了 tensorBoard 来可视化训练过程，用 tqdm 来显示训练的进度，同时设置了’early_stop’参数，当指定轮数没有改进时就停止训练。

三、实验结果

以下是数据预处理过程中的输出信息

```
各genre的统计
Drama                4361
Comedy               3756
Thriller             1894
Action               1828
Romance              1596
Adventure            1263
Crime                1199
Sci-Fi                980
Horror                978
Fantasy               779
Children              664
Animation             611
Mystery               573
Documentary           440
War                   382
Musical               334
Western               167
IMAX                  158
Film-Noir              87
(no genres listed)     34
dtype: int64
```

图 1 各 genre 的统计

```
class_counts:
3    4789
2    2596
4    1260
1     821
0     258
Name: rating_class, dtype: int64
class_proportions:
3    0.492493
2    0.266968
4    0.129576
1    0.084430
0    0.026532
Name: rating_class, dtype: float64
shape of average_ratings:(9724, 4)
shape of all_tags:(1572, 3)
```

图 2 各 rating 类别的个数和比例，average_ratings 和 all_tags 的 shape

```
含有缺失值的行号和movieId:
585       720
816      1076
2211     2939
2499     3338
2587     3456
3118     4194
4037     5721
4506     6668
4598     6849
4704     7020
5020     7792
5293     8765
5421    25855
5452    26085
5749    30892
5824    32160
5837    32371
5957    34482
7565    85565
Name: movieId, dtype: int64
共19条有缺失
shape of data:(9723, 34)
```

图 3 含有缺失值的行号和 movieId，初步预处理得到的 data 的 shape

以下是某次运行的结果：

```
shape of data:(9723, 34)
number of features: 33
逻辑回归模型：
模型准确率:0.566289825282631
分类报告：
              precision    recall  f1-score   support

       (0,1]       0.00      0.00      0.00        29
       (1,2]       0.30      0.07      0.12        80
       (2,3]       0.50      0.49      0.50       268
       (3,4]       0.60      0.87      0.71       464
       (4,5]       0.82      0.07      0.13       132

    accuracy                           0.57       973
   macro avg       0.44      0.30      0.29       973
weighted avg       0.56      0.57      0.50       973
```

图 4 逻辑回归模型结果

```
随机森林分类模型（优化前）：
模型准确率:0.5837615621788284
分类报告：
              precision    recall  f1-score   support

       (0,1]       0.25      0.07      0.11        29
       (1,2]       0.33      0.16      0.22        80
       (2,3]       0.53      0.53      0.53       268
       (3,4]       0.62      0.83      0.71       464
       (4,5]       0.68      0.20      0.31       132

    accuracy                           0.58       973
   macro avg       0.48      0.36      0.37       973
weighted avg       0.57      0.58      0.55       973
```

图 5 随机森林分类（优化前）模型结果

```
特征重要性分析：
Variable: star_level           Importance:0.1742
Variable: Users_reviews        Importance:0.0969
Variable: Critic reviews       Importance:0.0903
Variable: year                 Importance:0.0877
Variable: Director_stars       Importance:0.0749
Variable: Gross Worldwide      Importance:0.0714
Variable: Gross US/Canada      Importance:0.0666
Variable: Opening Weekend      Importance:0.0642
Variable: Budget               Importance:0.0578
Variable: rating_count         Importance:0.0496
Variable: time                 Importance:0.0152
Variable: Comedy               Importance:0.0146
Variable: Drama                Importance:0.0143
Variable: Action               Importance:0.0118
Variable: Romance              Importance:0.011
Variable: Thriller             Importance:0.0106
Variable: Crime                Importance:0.0101
```

图 6 随机森林分类（优化前）特征重要性分析（截取部分）

iter	target	max_depth	max_fe...	min_sa...	min_sa...	n_esti...
1	0.593	34.38	0.233	7.023	10.18	330.7
2	0.5971	14.71	0.6171	19.59	41.32	125.5
3	0.6002	28.89	0.6067	11.86	25.23	362.2
4	0.5899	28.2	0.5622	11.09	40.21	207.8
5	0.5797	42.05	0.2358	30.86	25.5	330.2
6	0.6002	49.98	0.6581	6.171	44.06	101.5
7	0.5838	62.55	0.2071	26.57	3.667	18.84
8	0.592	41.26	0.5613	19.31	49.73	87.42
9	0.592	89.17	0.5383	20.15	23.87	379.9
10	0.5868	64.13	0.2684	20.94	45.02	130.2
11	0.5982	82.85	0.4296	1.39	20.51	218.2
12	0.5992	43.85	0.6049	4.133	30.38	373.2
13	0.591	63.73	0.3373	10.07	34.69	288.2

图 7 贝叶斯优化过程

随机森林分类模型（优化后）：
模型准确率:0.593011305241521
分类报告：

	precision	recall	f1-score	support
(0,1]	0.33	0.14	0.20	29
(1,2]	0.42	0.20	0.27	80
(2,3]	0.55	0.53	0.54	268
(3,4]	0.62	0.83	0.71	464
(4,5]	0.70	0.21	0.33	132
accuracy			0.59	973
macro avg	0.53	0.38	0.41	973
weighted avg	0.59	0.59	0.56	973

图 8 随机森林分类模型结果

```
shape of data:(9723, 34)
number of features: 33
Epoch [1/100]: 100%|██████████| 547/547 [00:00<00:00, 1072.56it/s, loss=1.53]
Epoch [1/100]: Train loss: 1.3034, Valid loss: 1.3122
Saving model with loss 1.303...
Epoch [2/100]: 100%|██████████| 547/547 [00:00<00:00, 1183.77it/s, loss=1.22]
Epoch [2/100]: Train loss: 1.2841, Valid loss: 1.3134
Saving model with loss 1.284...
Epoch [3/100]: 100%|██████████| 547/547 [00:00<00:00, 1211.40it/s, loss=0.9]
Epoch [3/100]: Train loss: 1.2857, Valid loss: 1.3172
Epoch [4/100]: 100%|██████████| 547/547 [00:00<00:00, 1146.78it/s, loss=1.11]
Epoch [4/100]: Train loss: 1.2868, Valid loss: 1.2976
Epoch [5/100]: 100%|██████████| 547/547 [00:00<00:00, 1222.68it/s, loss=1.39]
Epoch [5/100]: Train loss: 1.2830, Valid loss: 1.3116
```

图 9 神经网络训练过程示意图

Model is not improving, so we halt the training session.
模型准确率:0.47687564234326824
分类报告：

	precision	recall	f1-score	support
(0,1]	1.00	0.00	0.00	29
(1,2]	1.00	0.00	0.00	80
(2,3]	1.00	0.00	0.00	268
(3,4]	0.48	1.00	0.65	464
(4,5]	1.00	0.00	0.00	132
accuracy			0.48	973
macro avg	0.90	0.20	0.13	973
weighted avg	0.75	0.48	0.31	973

图 10 神经网络分类模型结果

```
shape of data:(9723, 34)
number of features: 33
线性回归模型：
mse:0.5139219406299508, rmse:0.7168834916706834
accuracy(用1-mse表示):0.4860780593700492
随机森林回归模型（优化前）：
mse:0.5188742501738424, rmse:0.7203292651099512
accuracy(用1-mse表示):0.48112574982615763
特征重要性分析：
Variable: star_level           Importance:0.3954
Variable: Users_reviews        Importance:0.092
Variable: year                 Importance:0.08
Variable: Critic reviews       Importance:0.073
Variable: Director_stars       Importance:0.0565
Variable: Gross Worldwide      Importance:0.0522
Variable: Gross US/Canada      Importance:0.0428
Variable: Budget               Importance:0.0419
Variable: Opening Weekend      Importance:0.0374
Variable: rating_count         Importance:0.0298
```

图 11 线性回归和随机森林回归（优化前）模型结果

```
| 34        | 0.4994    | 69.64     | 0.4203    | 2.726     | 40.56     | 291.6     |
| 35        | 0.4929    | 68.12     | 0.9289    | 5.344     | 34.91     | 277.6     |
| 36        | 0.4973    | 75.84     | 0.5645    | 3.286     | 33.07     | 303.4     |
| 37        | 0.4923    | 36.87     | 0.8701    | 27.89     | 31.63     | 306.0     |
| 38        | 0.4476    | 51.03     | 0.1       | 31.9      | 31.83     | 239.5     |
| 39        | 0.496     | 64.44     | 0.5933    | 11.19     | 42.25     | 300.2     |
| 40        | 0.4916    | 44.52     | 0.9134    | 8.702     | 34.88     | 228.5     |
=================================================================================
{'target': 0.49937254993775115, 'params': {'max_depth': 69.63990670395819, 'max_features': 0.4202877888063936, 'min_samples_leaf': 2.7259638479282855,
 'min_samples_split': 40.560964783257134, 'n_estimators': 291.5712923483118}}
随机森林分类模型（优化后）：
mse:0.5020928783044307, rmse:0.7085851242472077
accuracy(用1-mse表示):0.4979071216955693
特征重要性分析：
Variable: star_level           Importance:0.4988
Variable: Users_reviews        Importance:0.0709
Variable: year                 Importance:0.0608
Variable: Critic reviews       Importance:0.0597
Variable: Director_stars       Importance:0.0591
Variable: Opening Weekend      Importance:0.0491
```

图 12 贝叶斯优化示意图，随机森林回归（优化后） 结果

```
Epoch [30/100]: 100%|██████████| 547/547 [00:01<00:00, 428.94it/s, loss=0.658]
Epoch [31/100]:   0%|          | 0/547 [00:00<?, ?it/s, loss=0.576]Epoch [30/100]: Train loss: 0.7606, Valid loss: 0.7734
Epoch [31/100]: 100%|██████████| 547/547 [00:01<00:00, 416.91it/s, loss=0.878]
Epoch [32/100]:   0%|          | 0/547 [00:00<?, ?it/s, loss=0.848]Epoch [31/100]: Train loss: 0.7623, Valid loss: 0.8028
Epoch [32/100]: 100%|██████████| 547/547 [00:01<00:00, 416.77it/s, loss=0.957]
Epoch [32/100]: Train loss: 0.7734, Valid loss: 0.7728

Model is not improving, so we halt the training session.
mse:0.6168976426124573, rmse:0.7854283179338883
accuracy(用1-mse表示):0.3831023573875427
```

图 13 神经网络回归模型结果

下表给出三次运行各模型的结果：

模型	第一次	第二次	第 3 次	平均值
逻辑回归	准确率 0.5663	准确率 0.5488	准确率 0.5858	准确率 0.5670
随机森林分类（优化前）	准确率 0.5838	准确率 0.5734	准确率 0.5910	准确率 0.5827
随机森林分类（优化后）	准确率 0.6043	准确率 0.5807	准确率 0.6002	准确率 0.5951
神经网络分类	准确率 0.4769	准确率 0.5087	准确率 0.5087	准确率 0.4981
线性回归	mse:0.5139 rmse:0.7169 准确率:0.4861	mse:0.4907 rmse:0.7005 准确率:0.5093	mse:0.4749 rmse:0.6891 准确率:0.5251	mse:0.4931 rmse:0.7022 准确率:0.5068
随机森林回归（优化前）	mse:0.5189 rmse:0.7203 准确率:0.4811	mse:0.5205 rmse:0.7215 准确率:0.4795	mse:0.4701 rmse:0.6892 准确率:0.5250	mse:0.5032 rmse:0.7103 准确率:0.4952
随机森林回归（优化后）	mse:0.5021 rmse:0.7086 准确率:0.4979	mse:0.4865 rmse:0.6975 准确率:0.5135	mse:0.4600 rmse:0.6783 准确率:0.5400	mse:0.4829 rmse:0.6948 准确率:0.5171
神经网络回归	mse:0.6169 rmse:0.7854 准确率:0.3831	mse:0.7475 rmse:0.8646 准确率:0.2525	mse:0.6679 rmse:0.8172 准确率:0.3321	mse:0.6774 rmse:0.8224 准确率:0.3226

可以看到经过优化后的随机森林算法在分类和回归问题中的效果都更好。此外在实验过程中发现神经网络的模型随着训练轮数的增加反而会出现不收敛甚至发散的现象，通过调参并未很好的解决该问题，可能是预处理的数据本身还不太适合使用神经网络模型，导致神经网络模型的效果甚至不如最简单的逻辑回归和线性回归模型。

通过查看随机森林的特征重要性分析结果，可以发现 imdb 的评分与 Movie Lens 数据集处理得到的评分相关性最大，这也很符合人的认知。此外，相关性较大的还有 User_reviews 数，Critic reviews 数，电影的发行年份，导演的导演过的电影的 imdb 平均星级，三种票房，预算。所有 genre 的和也占有较大的比例。当取消掉 imdb 评分作为特征时，各个模型的准确率会下降 0.05 左右。若将爬取的数据全部去掉，各个模型的准确率会下降 0.1 左右。

对于分类模型，相较于随机猜类别 0.25 的准确率，3 中模型的准确率都有较大提升，且随机森林分类的准确率可达 0.6，模型训练还是由较大的效果。

此外，当减少样本数，比如对含缺失值的样本直接删除而不做任何填充，样本数会减少一半（票房、预算等缺失的较多）,但各模型的准确率都能提升 0.05 左右。

置于准确率再难有较大提升，应该是数据本身的问题。比如给出 100836 各评分，平均每部电影约有 10 个评分，但实际数据中大量的电影的本分数很少，有的只有一两个，有的甚至没有评分，这导致处理作为标签的评分带有较强的主观性，并不能很好的反应一部电影的品质或受欢迎程度。

四、实验小结

在本次实验中，我首先进行了详细的数据预处理，包括从 Movie Lens Dataset 中提取电影属性，爬取外部数据，处理缺失值等。然后搭建了六个模型，包括逻辑回归、随机森林、神经网络等，通过多次运行获得了它们在分类和回归任务上的性能指标。实验过程中学到了数据清洗和模型选择的重要性，同时发现了神经网络模型在当前任务上表现不如传统机器学习模型。此外，也掌握了一些爬虫的基本技巧、贝叶斯优化和一些基础的多线程优化。总体而言，实验让我综合运用课程上所学到的知识，更加深入地了解 python 在大数据与人工智能上运用，同时也明白了数据的质量对模型性能的影响，为进一步的研究提供了指导。